

This page Is Inserted by IFW Operations
And is not part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of
The original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

**As rescanning documents *will not* correct images,
Please do not report the images to the
Image Problem Mailbox.**

THIS PAGE BLANK (USPTO)

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 08-207363

(43)Date of publication of application : 13.08.1996

(51)Int.Cl. B41J 5/30
G06T 11/00
H04N 1/41
// G06T 9/00

(21)Application number : **07-214558**
(22)Date of filing : **23.08.1995**

(71)Applicant : **CANON INC**
(72)Inventor : **SUGIURA SUSUMU
SAITO KAZUHIRO
TODA YUKARI**

(30)Priority

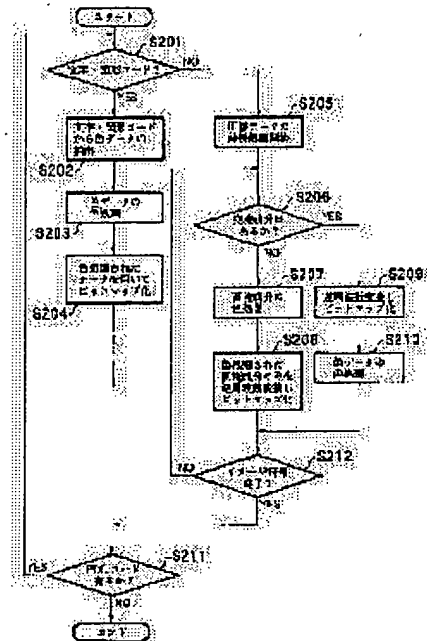
Priority number : 06207488 Priority date : 31.08.1994 Priority country : JP
06296992 30.11.1994 JP

(54) PICTURE IMAGE PROCESSING DEVICE AND METHOD

(57)Abstract:

PURPOSE: To realize the efficient conversion to the data dependent upon a visible image outputting device by a method wherein the image information written in the predetermined language is expanded into image data of every picture element and then outputs to a first output device so as to use a parameter, which is determined in response to both the characteristics of the first output device and those of a second output device.

CONSTITUTION: By recognizing a header command, inputted PDL code is judged to be whether a character and figure code or an image and mark code at the step S201. When the inputted PDL code is the character and figure code, color information is extracted from the code so as to color-process by a color processing parameter in response to an output device in order to expand image data of every bit-mapped image by means of the color-processed data at the steps S202-S204. Next, the presence of PDL code is judged at the step S211. When PDL code is present, the procedure returns to the step S201. When an PDL code is present, the procedure is brought to an end. Since color processing is executed before the pit-mapping of the color information of the PDL code, objects having the same color can be realized by one color processing, resulting in shortening the time required for color processing remarkably.



LEGAL STATUS

(19)日本国特許庁(JP)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平8-207363

(43)公開日 平成8年(1996)8月13日

(51)IntCl. ⁶	識別記号	庁内整理番号	F I	技術表示箇所
B 4 1 J 5/30	C			
G 0 6 T 11/00				
H 0 4 N 1/41	C			
		9365-5H	G 0 6 F 15/ 72	A
			15/ 66	3 3 0 B

審査請求 未請求 請求項の数32 OL (全 25 頁) 最終頁に続く

(21)出願番号 特願平7-214558

(22)出願日 平成7年(1995)8月23日

(31)優先権主張番号 特願平6-207488

(32)優先日 平6(1994)8月31日

(33)優先権主張国 日本(JP)

(31)優先権主張番号 特願平6-296992

(32)優先日 平6(1994)11月30日

(33)優先権主張国 日本(JP)

(71)出願人 000001007
キヤノン株式会社
東京都大田区下丸子3丁目30番2号

(72)発明者 杉浦 進
東京都大田区下丸子3丁目30番2号 キヤ
ノン株式会社内

(72)発明者 斎藤 和浩
東京都大田区下丸子3丁目30番2号 キヤ
ノン株式会社内

(72)発明者 戸田 ゆかり
東京都大田区下丸子3丁目30番2号 キヤ
ノン株式会社内

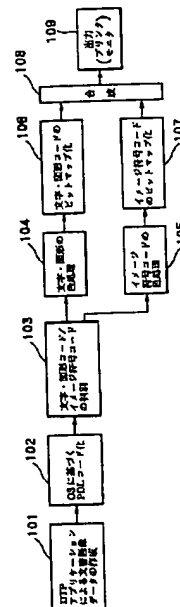
(74)代理人 弁理士 大塚 康徳 (外1名)

(54)【発明の名称】 画像処理装置及び方法

(57)【要約】

【課題】 色指定された文字コードや線画等を可視画像出力装置に出力する際、その可視画像出力装置に適応した色空間に効率良く、且つ高速に変換する。

【解決手段】 デスクトップパブリッシング等で作成した文書画像データを可視画像出力装置、例えば、カラープリンタに出力する場合には、PDLに基づくコードが送られてくる。このコードを受け、それが文字や図形である場合には、そのコードに含まれるRGBのカラーコードからYMCデータに変換する。そして、文字コードに基づいて発生した1ドット1ビットのデータに対し、その有意なビットをいっきにYMCデータに変換する。



1

【特許請求の範囲】

【請求項1】 所定言語で記述された画像情報を供給する供給手段と、

画像情報を画素毎の画像データに展開する展開手段と、
展開された画像データを第1の出力デバイスに出力する出力手段とを備え、

前記展開手段は、第1の出力デバイスの特性及び前記第1の出力デバイスとは異なる第2のデバイスの特性に応じて決定されるパラメータを画像情報を展開するために使用することを特徴とする画像処理装置。

【請求項2】 前記所定言語はページ記述言語であることを特徴とする請求項第1項に記載の画像処理装置。

【請求項3】 前記展開手段は中央演算処理装置であることを特徴とする請求項第1項に記載の画像処理装置。

【請求項4】 前記第1あるいは第2の出力デバイスはカラープリンタであることを特徴とする請求項第1項に記載の画像処理装置。

【請求項5】 前記第1あるいは第2の出力デバイスはカラーモニターであることを特徴とする請求項第1項に記載の画像処理装置。

【請求項6】 前記パラメータは色変換係数であることを特徴とする請求項第1項に記載の画像処理装置。

【請求項7】 前記パラメータは複数の関数の合成関数であることを特徴とする請求項第1項に記載の画像処理装置。

【請求項8】 前記第1又は第2の出力デバイスの特性は、色再現特性であることを特徴とする請求項第1項に記載の画像処理装置。

【請求項9】 前記第1又は第2の出力デバイスの特性は、通信ネットワークを介して受信されることを特徴とする請求項第1項に記載の画像処理装置。

【請求項10】 所定言語で記述された画像情報を供給する供給工程と、

画像情報を画素毎の画像データに展開する展開工程と、
展開された画像データを第1の出力デバイスに出力する出力工程とを備え、

前記展開工程では、第1の出力デバイスと前記第1の出力デバイスとは異なる第2の出力デバイスの特性に基づいて決定されるパラメータを、画像情報の展開に使用することを特徴とする画像処理方法。

【請求項11】 符号化画像情報を供給する供給手段と、

符号化画像情報を復号化し、画素毎の画像データを生成する復号手段と、

画像データを所定の出力デバイスに出力する出力手段と、

画像データが所定の出力デバイスの特性に依存するように演算する演算手段とを備え、

前記演算手段は、前記符号化画像情報のそれぞれのデータに対して演算することを特徴とする画像処理装置。

2

【請求項12】 前記符号化画像情報は、ブロック単位で直交成分と交流成分とを分離して符号化された画像データをであることを特徴とする請求項第11項に記載の画像処理装置。

【請求項13】 前記復号化手段はマイクロプロセッサによる処理で行うことを特徴とする請求項第11項に記載の画像処理装置。

【請求項14】 前記所定の出力デバイスはカラープリンタであることを特徴とする請求項第11項に記載の画像処理装置。

【請求項15】 前記所定の出力デバイスはカラーモニターであることを特徴とする請求項第11項に記載の画像処理装置。

【請求項16】 前記演算手段は、色変換係数に基づいて実行されることを特徴とする請求項第11項に記載の画像処理装置。

【請求項17】 前記それぞれのデータは、ブロック毎の直流成分であることを特徴とする請求項第11項に記載の画像処理装置。

【請求項18】 前記所定の出力デバイスの特性は色再現特性であることを特徴とする請求項第11項に記載の画像処理装置。

【請求項19】 前記所定の出力デバイスの特性は通信ネットワークを介して受信されることを特徴とする請求項第11項に記載の画像処理装置。

【請求項20】 符号化画像情報を供給する供給工程と、

符号化画像情報を復号化し、画素毎の画像データを生成する復号工程と、

画像データを所定の出力デバイスに出力する出力工程と、

画像データが所定の出力デバイスの特性に依存するように演算する演算工程とを備え、

前記演算工程は、前記符号化画像情報のそれぞれのデータに対して演算することを特徴とする画像処理方法。

【請求項21】 所定言語で記述された画像情報を供給する供給手段と、

画像情報を画素毎の画像データに展開する展開手段と、
所定の出力デバイスに画像データを出力する出力手段と

を備え、

前記展開手段は、前記所定の出力デバイスから受信した、当該所定の出力デバイスの特性に基づいて決定されるパラメータを使用することを特徴とする画像処理装置。

【請求項22】 前記所定言語はページ記述言語であることを特徴とする請求項第21項に記載の画像処理装置。

【請求項23】 前記展開手段はマイクロプロセッサによって実行されることを特徴とする請求項第21項に記載の画像処理装置。

3

【請求項24】 前記所定の出力デバイスはカラープリンタであることを特徴とする請求項第21項に記載の画像処理装置。

【請求項25】 前記所定の出力デバイスはカラーモニターであることを特徴とする請求項第21項に記載の画像処理装置。

【請求項26】 前記パラメータは色変換係数であることを特徴とする請求項第21項に記載の画像処理装置。

【請求項27】 前記パラメータはマトリクスによって表されることを特徴とする請求項第21項に記載の画像処理装置。

【請求項28】 前記所定の出力装置の特性は色再現特性であることを特徴とする請求項第21項に記載の画像処理装置。

【請求項29】 前記所定の出力装置の特性は通信ネットワークを通して受信することを特徴とする請求項第21項に記載の画像処理装置。

【請求項30】 所定言語で記述された画像情報を供給する供給工程と、
画像情報を画素毎の画像データに展開する展開工程と、
所定の出力デバイスに画像データを出力する出力工程とを備え、
前記展開工程は、前記所定の出力デバイスから受信した、当該所定の出力デバイスの特性に基づいて決定されるパラメータを使用することを特徴とする画像処理方法。

【請求項31】 所定の言語で記述された文書画像データを、可視画像出力装置に出力する画像処理装置であって、
文書画像データ中に文字コードあるいは図形描画指示コードと、その出力色指定コードがあるかどうかを判断する判断手段と、

文字コードあるいは図形描画指示コードと、その出力色指定コードがあると判断した場合、当該出力色指定コードに基づく前記可視画像出力装置の色空間データでもって、可視画像用ビットマップデータを展開し、出力する制御手段とを備えることを特徴とする画像処理装置。

【請求項32】 圧縮画像データを伸長して出力デバイスへ出力する画像処理装置であって、
前記出力デバイスの特性に基づいて前記圧縮画像データの代表色表現データを処理する処理手段と、
前記処理手段によって処理されたデータを用いて前記圧縮画像データのコードをビットマップデータに伸長する伸長手段とを有することを特徴とする画像処理装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 本発明は画像処理装置及び方

4

法、詳しくは入力した文書画像データを可視画像出力装置に出力するための画像処理装置及び方法に関するものである。

【0002】

【従来の技術】 DTP (Desk Top Publishing) で作成された文書画像データは、図13で示されるように何等かのPDLで記述され、ラスタ化し、出力プリンタまたは出力モニタの色特性に合わせる色処理、すなわち、CMM (Color Matching Method) 処理を施し、出力部に渡されプリント、またはモニタ表示される。従来の方法は、ラスタ化した後、1画素毎に色処理計算を施し、全画素にわたり色合わせ処理を実施していた。

【0003】 また、出力プリンタや出力モニタの色特性は、装置毎に異なっているため、例えば、1つのホストコンピュータに対して複数の出力プリンタや出力モニタはネットワークで接続された場合に、いかなる色特性で色合わせを行ったら良いのかわからないという問題があった。

【0004】 また、一方で、コンピュータで作成する画像（以下「CG画像」という）のカラー化、多階調化が進んでいる。このような多色画像の情報量は、例えば、A4サイズ400dpi256階調三色カラーの場合で約46Mバイトにもなる。従って、このような画像を格納したり伝送する際には、画像圧縮することになる。このような圧縮された画像を伸長して、プリンタまたはディスプレイなどのデバイスに出力する際は、その出力デバイスに適した色変換またはγ変換を行う必要がある。図14は圧縮データを伸長してカラープリンタに出力する場合を説明する図である。

【0005】 図14において、入力された圧縮データは、符号解析部901Aと画素データ発生部901Bを備える伸長部901においてRGB各8ビットの画素データに変換された後、色合部902で色合わせされてCMYK各8ビットの画像データに変換され、カラープリンタ903へ渡される。図15はこの色合部902の処理を説明する図である。

【0006】 図15において、まず、1011で示すLOG変換処理において次式により、RGB色空間からYMC色空間への変換が施される。

【0007】

$$\begin{aligned} Y &= -\log B \\ M &= -\log G \\ C &= -\log R \end{aligned} \quad \dots(1)$$

次に、1012で示すマスキング処理において次式により、プリンタ特有の色特性に合わせるマスキング処理がYMC信号に施される。

【0008】

$$\begin{bmatrix} Y \\ M \\ C \end{bmatrix} = \begin{bmatrix} a11 & a12 & a13 \\ a21 & a22 & a23 \\ a31 & a32 & a33 \end{bmatrix} \cdot \begin{bmatrix} Y \\ M \\ C \end{bmatrix} \quad \dots(2)$$

さらに、1013で示す黒生成処理により、図16に示すように、Y'M'C'信号の最小値、つまり値 $\min(Y', M', C')$ をY'M'C'信号から引き、その分をK'信号に置き換える処理が施される。

【0009】このように、伸長された画素データを、色変換マトリクスなどを用いて、一画素ずつ変換している。

【0010】しかし、上記従来例においては、次のような問題点があった。

【0011】つまり、上述した技術においては、伸長された画像の画素データすべてについて色処理演算を施す必要があり、その処理時間に莫大な時間を必要とする。もし、処理時間を短縮しようとするれば、高速演算を実行できるハードウェアが必要になり、装置コストを上昇させる欠点がある。

【0012】

【発明が解決しようとする課題】本発明はかかる従来技術の欠点を除去することを目的とする。

【0013】すなわち、本発明は、色指定された文字コードや線画等の画像情報を可視画像出力装置に出力する際、その可視画像出力装置に依存したデータに効率良く、且つ高速に変換することを可能ならしめる画像処理装置及び方法を提供しようとするものである。また、その際に、複数の出力装置間の色合わせを行うことを目的とする。

【0014】この課題を達成するため、例えば本発明の画像処理装置は以下の構成を備える。すなわち、所定言語で記述された画像情報を供給する供給手段と、画像情報を画素毎の画像データに展開する展開手段と、展開された画像データを第1の出力デバイスに出力する出力手段とを備え、前記展開手段は、第1の出力デバイスの特性及び前記第1の出力デバイスとは異なる第2の出力デバイスの特性に応じて決定されるパラメータを画像情報を展開するために使用する。

【0015】また、本発明は、所定画素ブロック単位の符号化画像データがある場合において、可視画像出力装置に依存したデータを効率良く且つ高速に変換することを可能ならしめる画像処理装置及び方法を提供することを別の目的とする。

【0016】このため、例えば本発明の画像処理装置は以下の構成を備える。すなわち、符号化画像情報を供給する供給手段と、符号化画像情報を復号化し、画素毎の画像データを生成する復号手段と、画像データを所定の出力デバイスに出力する出力手段と、画像データが所定の出力デバイスの特性に依存するように演算する演算手

段とを備え、前記演算手段は、前記符号化情報のそれぞれのデータに対して演算する。

【0017】また、本発明は、実質的に同じ色あいのカラー画像に関して、可視画像出力装置に適應するデータに変換する処理を実質的に行わず、効率良くその装置に適應したデータを生成することを可能ならしめる画像処理装置及び方法を提供することを更なる目的とする。

【0018】また、本発明は、圧縮された画像を伸長して出力デバイスへ出力する際の処理を低減することができる画像処理装置およびその方法を提供することを他の目的とする。

【0019】また、本発明はネットワーク上に接続された複数のデバイスを用いた効率の良い画像処理方法を提供することを他の目的とする。

【0020】本発明の他の目的及び形態は以下の図面に基づく説明及び特許請求の範囲の記載から明らかになるであろう。

【0021】

【発明の実施の形態】以下、添付図面に従って本発明に係る好適な実施形態を詳細に説明する。

【0022】＜第1の実施形態＞図1は、本発明の実施形態におけるデータの流れを示していると同時に、具体的な構成名は記載してはいないが実施形態の装置構成をも示している。以下、図1の説明を行う。

【0023】101にて、コンピュータのDTPアプリケーションソフトウェアにより、文書画像データを作成する。次に102にて、作成された文書画像データをOSに基づいてPDLコードに変換する。そして、103にて、文字図形コードとイメージ符号コードの判別を行い、文字図形コードとイメージ符号コードは、それぞれ別々に処理される。文字図形は、104にて文字図形用の色処理を行い、106にて、その色処理された画素データを用いてビットマップ化する。また、イメージ符号コードは、105にてイメージ符号コード用の色処理を行い、107にて、その色処理されたデータを用いてビットマップ化する。108では、文字図形コードとイメージ符号コード、別々に色処理、ビットマップ化されたデータを合成する。109では、合成された画像をプリント或いはモニタ表示する。

【0024】尚、上記処理において、モニタに表示するデータはRGB形式になり、プリンタに出力するデータはYMC形式（あるいはBk成分を追加した形式）になるので、出力対象に応じて106、107の色処理内容を変化させる。

【0025】図2は本発明の特徴を最もよく表すフロー

7

チャートであり、PDLコードから色処理してビットマップ化するまでの本発明に係る処理を明示している。また、イメージ符号コードは、自然画像等の画像データが、周波数変換処理を含む圧縮方式を用いて、符号化されていることを前提にしている。本フローチャートでは、本発明に関連するその周波数変換関連の処理を明示し、それ以外の処理に関しては、省略している。文字・図形・イメージ符号コードの各々の色処理は、後から別々に詳しく説明する。

【0026】以下、この図2を用いて本発明の特徴を説明する。ステップS201にて、入力されるPDLコードが、文字図形コードかイメージ符号コードか判定する。この判定は、ヘッダコマンドを識別することで行う。つまり、イメージ符号コードの場合には、少なくともそのヘッダ部分に所定形式のコマンドが付されているので、これを識別する。また、文字図形コードはそのコードを解析することで識別する。文字図形コードならば、ステップS202にて、そのコード（あるいはコマンド）から色情報を抽出する。そして、ステップS203にて、出力装置に応じた色処理パラメータにより色処理を行い、ステップS204にて、その色処理されたデータを用いてビットマップ化画像毎のイメージデータに展開する。次に、ステップS211にて、PDLコードがあるかどうか判定し、Yesならば、ステップS201に戻り、Noならば終了する。

【0027】従来の方式では、PDLコードがビットマップ化された後で色処理するので、ビットマップ化された全ての画素に対して、色処理しなければならず、その計算に多くに時間を要した。しかし、本実施形態では、PDLコードの色情報をビットマップ化する前に色処理するため、同色のオブジェクトに対しては一回の色処理で実現することができ、大幅に色処理に費やす時間を短縮することができる。

【0028】さて、入力されたPDLコードが、イメージ符号コードの場合、ステップS201ではNoと判定され、ステップS205に進む。ここでは、圧縮データの伸張処理を始める。周波数変換を含む圧縮方式で圧縮データは、一般的に $n \times n$ （又は $n \times m$ ）のブロック単位で処理をする。ステップS206では、その周波数空間に変換された単位ブロックにおいて交流成分を含むかどうか判定し、含む場合は、従来通り処理するためのステップS209にて、逆周波数変換しビットマップ化し、ステップS210にて、ビットマップ化された全ての色データを出力装置に応じて色変換する。

【0029】また、ステップS206にて、交流成分がないと判定されたならば、ステップS207にて、その単位ブロックの直流成分のみを用いて出力装置に応じて色処理する。そして、ステップS208にて、色処理された直流成分のみを逆周波数変換しビットマップ化する。ステップS211では、このイメージ符号が終了し

8

たかどうかを判定し、終了した場合は、ステップS211へ進み、終了していない場合は、ステップS206へS210の処理を繰り返す。

【0030】従来の方式では、無条件で逆周波数変換し、ビットマップ化した後で出力装置の色特性に合わせて色処理を行っていたため、全ての画素に対しその計算を行わなければならなかった。しかし、本発明では、自然画の背景等は、交流成分をあまり含まず直流成分のみの領域が多いこと特徴を利用して、直流成分のみを色変換する異ことにより、大幅に色処理のための計算時間を短縮できる。

【0031】以上、本実施形態の特徴とする部分に関して述べたが、以下に文字コード、図形（グラフィック）コード、イメージ符号コードの各々の色処理に関して具体例を示しながら説明する。

【0032】（1）文字コードデータから効率よく色合わせ処理する方法。

【0033】説明を簡潔で具体的にするために、一枚の紙の上にピンク色の文字“東京”と、薄い青色の図形“円”を各例を基準に説明する。

【0034】図3は上記説明例を図示したものである。この図の或種のPDLコードを文字の場合

ESC[001 =moji, xadr, yadr, Rdata, Gdata, Bdata, size, font, char]

ESC[002 =circle, xadr, yadr, Rdata, Gdata, Bdata, radius]

ここで、ESCはエスケープコード（16進数で1Bh）、mojiは文字データを、circleは円を描かせるコマンド、xadr, yadr は文字の書き始め位置情報又は円の中心位置情報を表わす。また、Rdata, Gdata, Bdata は文字、円の色データ情報を示す。sizeは文字の大きさを示し、fontは文字フォントを示す。charは記載文字データ列を示す。radius は円描画の半径を示す。

【0035】従って、図3の場合、そのPDLデータは、

ESC[001, 0100, 0075, 250, 010, 025, 030, 001, 東京]

ESC[002, 0180, 0300, 010, 020, 0250, 030]

で表現される。

【0036】尚、実用的なPDLはさらに複雑な形式になっているが、ここでは本発明の思想が理解できる簡便な形に簡略化している。通常PDLに記載されている画像表現データはデバイスインデペンデントな形式で記述されている。ここでは、デバイスインデペンデントな色表現形式としてNTSC-RGBを用いたとする。出力装置もここでは、カラープリンタと設定すると、カラープリンタは減色混合であるからYMC（yellow, magenta, cyan）色空間となる。そのためNTSC-RGB→YMC色空間変換を実施する必要がある。

【0037】図4において、401は前記PDLの文字コードを蓄積しているデータメモリである。402は文

字コードからビットマップデータに変換する文字生成器である。403はPDL文字データRdata, Gdata, Bdataを抽出し出力のための色変換を実施する部分である。変換係数a11~a33は標準色空間NTSC-RGBから出力カラープリンタの色空間YMCに変換する変換係数である。404は402により文字コードデータを単色のビットマップデータに変換されたデータを格納する部分である。405は404でビットマップ化されたデータに色付ける部分である。色付けデータは403から生成されるY, M, Cデータを用いる。

【0038】405-1~3は各Y, M, C成分データ格納領域を示す。この場合のR, G, Bデータ値はR=250, G=10, B=25となる。

【0039】例えば、文字生成器402で生成された文字パターンは1画素当たり1ビットのデータが割り当てられていて、有意なドット(ビット)は“1”、非有意なドットは“0”になる。そこで、この有意なドット“1”に対しては、図示の行列式で得られたY, M, Cの各データ(例えば各色成分8ビット)の値を対応させ、それぞれの色成分データ格納領域に格納する。尚、20 非有意のドット“0”に対しては、その背景色(背景色はベクトル制御コードで指定されている)の各色成分のデータを格納する。そして、少なくとも1つの文字パターンに関しては無条件に、図示の行列演算を行うことなく、上記処理を行う。

【0040】以上の説明から明かなように、“東”の文字に関する色処理は1回ですむ。従来の方式では1文字を25×25ドットで表示していると625回色処理計算を繰り返す必要があった。上記実施形態の場合、少なくとも繰り返し回数は従来の1/625になるわけであ 30 るから、どの程度処理速度が改善されたか理解できよう。

【0041】(2) 図形(グラフィック)コードデータから効率よく色合わせ処理する方法。

【0042】図5は円画像を描画する例を説明する図である。501はベクトルデータが格納されている。この場合は円を描画するコマンドが格納されている。円を描画するベクトルデータとし、円のコマンド(その中心座標データ)と半径データが502に渡され、502でビットマップ化される。ここでは文字データの展開と同様 40 に、504には1ビットデータとして、データ格納される。一方、ベクトルの色情報を示すデータが503に渡され、色変換処理される。ここではR=10, G=20, B=250で半径は30と与えられる。従って、文字部と同様に、505には503で色処理計算されたデータY, M, Cが504の情報と合わせ格納される。従って、ここでも色処理計算は1回でよい。従来の方式では64×64ドットで表現されたとすると実に4069回の色処理計算が必要となる。

【0043】(3) イメージ符号コードから効率よく色 50

合わせ処理する方法。

【0044】イメージ符号コードは、自然画等の生画像を圧縮処理してイメージ符号のコードの集まりにしたものである。それを伸張しながら効率よく色合わせ処理する方法を説明する前に、先ず、その圧縮処理について簡単に説明する。これにより、その後の本発明の実施形態の記述がよりよく理解できるものと考えられる。

【0045】図6は、その周波数変換を含む圧縮処理の概要を示すブロック図である。生画像データは、周波数変換部601にて、(n×n)PIXELのブロック単位で、例えばDCT変換等の直交変換を用いて周波数空間に変換され、その変換された値は、周波数変換係数と呼ばれる。量子化部602では、同様にn×nブロック単位で量子化処理を行い、その量子化された値は、量子化係数と呼ぶ。量子化係数は、1個の直流成分と(n×n-1)個の交流成分で構成される。直流成分は、ブロック遅延部603にて、遅延され前のブロックとの差分が、符号化部604にて、符号化される。又、その他の交流成分は、いわゆるジグザグスキャンされ符号化部604にて符号化される。

【0046】次に、その画像データの変換される様子を図7~9に示される具体例を用いて説明する。図7は、(n×n)画素幅の画像データで、左下方向への斜線で示される領域Aは、全てAの値の色であり、右下方向への斜線で示される領域Bは、全てBの値の色である。領域Aと領域Bは、ちょうどn×nブロックの境界線で区切られているため、最初の2nラインまでは、交流成分が発生しない。図7を周波数変換して量子化したデータを示したものが図8である。図8は、生画像データの表現形式が、NTSC-RGBが用いられ、そのR, G, B-Planの量子化係数が表示され、最全面(B-Plane)の左上のn×nブロックにおいて、左上隅の20は直流成分を示し、0はその他の交流成分が、全て0であることを示す。図9上は、そのRGB直流成分の変遷を示し、図9下は、RGB直流差分成分の変遷を示す。

【0047】さて、次に、実施形態における符号化圧縮データを伸張しながら、出力装置の色特性に合わせて効果的に色処理する方法を図10を用いて説明する。

【0048】圧縮されたイメージ符号コードは、復号化部1001にて、復号化され、n×nブロック単位で量子化係数が、交流成分検出部1002へ転送される。交流成分を含む場合は、従来の伸張色処理を行い、交流成分がない場合(交流成分の値が全て“0”の場合)は、本発明の実施形態の伸張色処理を行う。すなわち、交流成分を含む場合、その直流成分は、1ブロック前の差分データであるため、ブロック遅延部1008を用いて1ブロック前の値との加算値が、交流成分はそのまま逆量子化部1009にて逆量子化される。逆量子化された係数は、逆周波数変換部1000にて逆数は数成分に変換され、色処理部1011にて出力装置の1007の色特

11

性に応じて、全ての画素が、色処理される。

【0049】一方、交流成分を含まない場合は、差分直流成分のみが、ブロック遅延部1003を用いて、1ブロック前の値との加算値が、色処理部に送られる。色処理部1004では、その加算された直流成分を出力装置の色特性に応じて色処理される。また、この色処理部1004は、交流成分検出部1002からの差分直流成分が、0である場合は、その加算された直流成分は、1つ前のブロックの値と同じなので、色合わせのための計算は、行わずに1ブロック前の値を用いる。この処理により、より効率的な色処理を実現することができる。そして、色処理されたデータは、逆量子化部1005、逆周波数変換部1006を経て、出力装置にて出力される。

【0050】ここでの色処理は、周波数変換された係数を色合わせのための計算を行うが、交流成分を含む場合は、その色変換のために複雑な計算を要する。しかし、交流成分がないため、その値は、逆周波数変換された値と線形関係にあり、簡単な計算で色合わせのための計算を実現することができる。

【0051】例えば、図7～9の例では、従来の方式で20ビットマップから色処理をした場合、最初の2nラインで(2n×5n)回の計算を要するが、図9下の図からわかるように異なる差分直流成分は、4回しか出現しない。つまり、実に4回の計算だけでこの領域における全ての色処理を実現することができことになる。

【0052】＜第2の実施形態＞第1の実施形態では、文字図形・イメージ符号のPDLコードを各々展開、或いは伸張する時に色処理をすることにより色合わせのための計算時間を大幅に短縮することを実現しているが、第2の実施形態では、生画像データを直接効率的に色処30理する例を述べる。

【0053】第2の実施形態における特徴を最もよく表わすフローチャートが図11である。実第2の実施形態では、いくつかの生画像データを保持するための保持手段とその生画像に対応する色処理された色データを保持するための手段を有している。すなわち、一度演算した色データについて、生画像データと色処理後のデータとの対応をルックアップテーブル化することにより、同一生画像データに対する重複演算を省略している。

【0054】さて、ステップS1101では、入力され40た生画像データが、保持されている生画像データと同一かどうか判定される。Noの場合は、ステップS1102にて、出力装置の色特性に応じた色合わせのための処理が成される。ステップS1103では、その生画像データは、生画像データ保持手段に保持され、その色処理されたデータは、色処理された色データ保持手段に保持される。そして、ステップS1104にて、色処理されたデータは出力される。

【0055】また、ステップS1101にて、入力された生画像データが、保持されている生画像データと同じ50

12

場合は、ステップS1105にて、保持されている生画像データに対応する色処理されたデータを出力し、この場合は、色処理のための計算は行われない。ステップS1106では、生画像データが終了したかどうか判定され、終了していない場合は、ステップS1101から1105までの処理を繰り返す。生画像データの終了と共にこの色処理も終了する。

【0056】対象となる画像データが、DTPソフト等で作成された場合、同じ画素値が連続したり、一度出現した画素値は、繰り返し出現する傾向がある。そのため、本発明を用いることにより、画素値が連続したり、再出現した場合には、改めて色処理のための計算をする必要がなく、その膨大な計算時間を大幅に短縮することができる。

【0057】以下、図12の画像データを例にして、その計算回数が削減される例を具体的に説明する。図12の1つの四角は、1画素を示し、16画素の幅の画像データが、A～Fまでの6ラインが示してある。空白の四角は、白色の画素であることを示し、「赤」は赤色の画素であることを、「青」は青色の画素であることを示す。ラスタ順次で画素データが入力されるとA行1列画素（以下、A-1画素と表記する。）である白色が色処理され、その後A-2, 3, は、その色処理されたデータが出力される。このあと、A-4の赤色が処理され、その後A-5, 6... 16, B-1, 2, 3... 16, C-1, 2, ... 10まで、保持された色処理後の色データが使われ、C-11の青色の画素データが、色処理される。その後、F-16までは、新たに色処理のための計算は行われず、保持された色処理後のデータが用いられる。従って、この例では、従来方式では、6×6=96回の色処理のための計算が必要になるが、本方式を用いることにより、たった3回の計算でこの領域における全ての画素の色処理を実現することができる。

【0058】尚、記憶保持するデータ数は、メモリ（例えばRAM）の容量に依存するものとする。実際問題として、DTPで作成した色数は、ビデオカメラ等で撮影した自然画と比較し、その色数は少ないので、適当なメモリ容量があれば足りる。

【0059】また、自然画の場合には、その色数が多くなるので、メモリの容量が許す限り活用することが望ましい。但し、新たに記憶保持するエリアが確保できない場合には、最も古い記憶保持したデータに上書きする。即ち、ルックアップテーブルを動的に書換えていく。いずれの場合にも、入力データに対し、すでに色処理されたデータが保持されている場合は、その保持されているデータを用い、保持されたデータが存在しない場合は、新たに色処理を行い、そのデータを保持するとともに、そのデータを出力する。

【0060】以上説明したように本第2の実施形態によ

13

れば、入力された画像データに対して色合わせ処理するとき、従前に処理したものと同一場合には、記憶しておいたデータを活用するので、その処理速度を大幅に向上させることが可能になる。

【0061】また、上記第1、第2の実施形態では、色合わせとしてRGBからYMCに変換する例を示したが、色空間の変換に特徴があるのであって、これ以外の色空間どうしの変換にも勿論対応できる。従って、上記実施形態によって本願発明が限定されるものではない。

【0062】また、第1の実施形態においては、伸長し10 た画素ブロックに交流成分が“0”になった場合、図2におけるステップS207、S208を行うとしたが、これにある程度の幅を持たせても良い。例えば、速度を重視したいのであれば、各交流成分（もしくはその平均）がT以下の場合にその処理を行うようにしても良い。但し、このTの値は適当に小さい値が望ましいのは説明するまでもないであろう。場合によっては、このTの値も操作者が自由に変更できるようにしても良いであろう（但し、デフォルトではT=0とする）。

【0063】以上説明したように上述の実施形態によれ20 ば、色指定された文字コードや線画等を可視画像出力装置に出力する際、その可視画像出力装置に適応した色空間のデータに効率良く、且つ高速に変換することが可能になる。

【0064】また、所定画素ブロック単位の周波数変換による符号化カラー画像データがある場合において、可視画像出力装置に適応した色空間のデータを効率良く且つ高速に変換することが可能になる。

【0065】また、上記2つの作用効果を同時に奏することも可能になる。

【0066】更にまた、実質的に同じ色あいのカラー画像に関して、可視画像出力装置に適応するデータに変換する処理を実質的に行わず、効率良くその装置に適応したデータを生成することが可能になる。

【0067】＜第3の実施の形態＞図17は本発明にかかる第2の実施形態の画像処理装置の構成例を示すブロック図で、圧縮された画像データの代表色を表現するデータにだけ、出力デバイスの特性に合せた変換処理を施し、変換処理したデータを用いて、圧縮コードをビットマップデータに伸長して、カラー画像データを出力デバ30 イスへ出力するものである。

【0068】より具体的に説明すると、図17において、入力された圧縮データは、符号解析部2011A、色合部2011Bおよび画素データ発生部2011Cを備える伸長部2011において、例えば、YMCK各8ビットの画素データに変換されてカラープリンタ2013などの出力デバイスへ渡される。画素データ発生部2011Cは、数色分の色データを記憶するバッファを備え、符号解析部2011Aの解析結果に従って、そのバッファの中身を更新する。その更新の際に、色処理部250

14

011Bにより色合わせを行い、色合わせされた色データをバッファに書込む。なお、ここでいう色合わせとは、出力デバイスの色特性に応じて行うもので、例えば、色空間変換、輝度-濃度変換、色再現範囲変換、マスキング、UCR、γ変換などの処理である。また、以下では、カラープリンタにCMYKデータを出力する例を説明するが、本実施形態はこれに限らず、例えば、カラーモニタなどの出力デバイスの場合には、RGBデータを出力すればよい。

【0069】次に、本第3の実施形態の処理を詳細に説明するが、まず、第3の実施形態が処理する圧縮データを説明する。なお、本実施形態が処理する圧縮データは、これに限るものではなく、同様の形態を備えた圧縮データであればよい。

【0070】図18は画像の一例を示す図で、128×128画素をもつRGB各8ビットのカラー多色画像で、Aで示す白地の上に、赤味を帯びた輪(B)、青味を帯びた四角形(C)および黒のライン(D)が描かれている。この画像を、ラスト順に、色データとそのランレングス（同色画素の連続数）に基づいて圧縮するが、ランレングスを最長255（つまり8ビットで表される）として、ここでは説明を簡単にするために、第64ラインの圧縮処理だけを説明する。

【0071】第64ラインは、図18に示すように、色Aが24画素、色Bが8画素、色Aが32画素、…、色Aが12画素で構成されている。ここで、各色データが次のように表されるとすると、その圧縮結果は図6のようになる。

【0072】色A: RGB=(255, 255, 255)

30 色B: RGB=(240, 64, 0)

色C: RGB=(128, 128, 255)

色D: RGB=(0, 0, 0)

図19において、一行32ビット中、最初の24ビットはRGBデータで、後の8ビットはランレングスである。つまり、図19の最初の行は色Aが24画素連続していることを示し、二行目は色Bが8画素分連続していることを示す。以下同様である。原画像の1ラインは3バイト×128画素で384バイトになるが、第64ラインを圧縮した結果は4バイト×9で36バイトになり、約十分の一に圧縮されたことになる。

【0073】図20はこのようにして圧縮されたデータを伸長して出力する処理手順の一例を示すフローチャートで、圧縮データが入力されると伸長部2011が実行するものである。

【0074】図20において、まず、ステップS2200で未処理の圧縮データがあるか否かを判定して、あればステップS2201へ進み、なければ処理を終了する。

【0075】未処理の圧縮データがあれば、ステップS2201で変数codeに一組の符号 図19の符号例では

15

32ビット)を読み、ステップS2202でその符号を解析する。つまり、ステップS2202では、変数codeを右に24ビットシフトしマスクデータFFHと論理積してRデータを取り出し、同様に、変数codeを右に16ビットシフトしFFHと論理積してGデータを取り出し、変数codeを右に8ビットシフトしFFHと論理積してBデータを取り出す。さらに、変数codeとFFHとを論理積してランレングスを取り出し、その値を変数timesへ格納する。なお、シフト量やマスクデータは、符号の形態、つまりRGBデータとランレングスの並び方およびビットサイズに応じて変化することはいうまでもない。

【0076】次に、ステップS2203で解析した符号の色データR, G, Bを色合わせを施して、その結果得られるY'M'C'K'データをバッファに格納する。具体的には、ステップS2203AでLOG変換によりRGBデータをCMYKデータに変換し、ステップS2203Bでマスキング演算によりCMYKデータを出力デバイスの特性に合わせたC'M'Y'データに変換し、ステップS2203CでUCRによりC'M'Y'データからその最小値 $\min(Y', M', C')$ を引くとともに、黒データK' 20を生成する。

【0077】続いて、ステップS2204で変数iに零をセットし、ステップS2205で変数iと変数timesを比較して、 $i < \text{times}$ であればステップS2206へ進み、そうでなければステップS2200へ戻る。ステップS2206ではバッファに格納したY'M'C'K'データを出力し、続くステップS2207で変数iをインクリメントしてステップS2205へ戻る。従って、ステップS2205からS2207のループが変数times分だけ繰り返され、バッファに格納されたY'M'C'K'データが変数times分だけ出力されることになる。そして、ループが変数times分だけ繰り返されると、処理はステップS2200へ戻り、未処理の圧縮データがあれば、次の一組の符号が処理されることになる。

【0078】このように、本実施形態においては、圧縮された画像データの代表色表現データにだけ、色合わせ処理を施し、処理した色データをランレングス分、つまり画素が連続する分、出力する。従って、伸長した画素データごとに色合わせ処理を行う必要がなくなり、画素ごとに色に変化するような画像でない限り、色合わせ処理に要する演算時間を短縮することができる。例えば図19に示す36バイトの圧縮データの場合、画素データごとに色合わせ処理を行うと128回の処理が必要になるのに対して、本実施形態においては9回の色合わせ処理で済み、その演算時間を約1/14に短縮することができる。さらに色数の少ないラインでは、より演算時間を短縮できることは明らかであり、一色で構成されるラインは一回の演算で済ませることができる。

【0079】<第4の実施形態>以下、本発明にかかる第4の実施形態の画像処理装置を説明する。なお、第450

16

実施形態において、第3実施形態と略同様の構成については、同一符号を付して、その詳細説明を省略する。

【0080】以下では、画像をパレットテーブルを用いて、ラスト順に、圧縮（以下「パレット圧縮」という）した圧縮データを伸長する例を説明するが、説明を簡単にするため、そのパレットを2ビット（4色）にする。しかし、カラー画像を実用的にパレット圧縮するには、少なくとも8ビット（128色）のパレットが必要になることはいうまでもない。

【0081】図21はパレット圧縮による画像データの圧縮手順例を示すフローチャートである。

【0082】まず、ステップS2301で設定済パレット色数を表す変数Xに零をセットし、ステップS2302で未処理の画素データがあるか否かを判定して、あればステップS2303へ進み、なければ処理を終了する。未処理の画素データがある場合はステップS2303で変数iに零をセットし、ステップS2304で変数iと変数xとを比較して、 $i < x$ であればステップS2305へ進み、そうでなければステップS310へ進む。

【0083】 $i < x$ の場合はステップS2305で画素データとパレットcol[i]とを比較して、画素データ=col[i]であればステップS308で圧縮コードcodeに変数iの値をセットした後、ステップS2302へ戻る。また、画素データ $\neq \text{col}[i]$ であればステップS2306で変数iをインクリメントした後、ステップS2304へ戻る。

【0084】また、ステップS2310へ進んだ場合は、パレットcol[x]に画素データをセットし、ステップS2311で圧縮コードcodeに変数xの値をセットし、ステップS2312で変数xをインクリメントした後、ステップS2302へ戻る。

【0085】図18に示す画像を図21に示す手順で圧縮すると、その第1画素においては、 $i=x=0$ なのでステップS2310へ進んで、パレットcol[0]に色Aのデータ(255, 255, 255)をセットし、ステップS2311で圧縮コードcodeに“00”をセットし、ステップS2312で変数xを1にした後、ステップS302へ戻る。

【0086】第2画素においては、ステップS2304で $i=0$ かつ $x=1$ であるからステップS2305へ進み、画素データはパレットcol[0]と一致するので、ステップS2308で圧縮コードcodeに“00”をセットした後、ステップS2302へ戻る。そして、しばらくの間は色Aが続き、その間、画素データとパレットcol[0]とは一致し、圧縮コードcodeは“00”になる。

【0087】続いて、色D(0, 0, 0)の画素に達すると、ステップS2305で画素データとcol[0]が不一致になり、ステップS2306で変数iが1になりステップS2310へ進んで、パレットcol[1]に(0, 0, 0)がセットされ、ステップS2311で圧縮コードcod

17

eに“01”がセットされ、ステップS2312で変数xは2になった後、ステップS2302へ戻る。

【0088】この処理を繰り返すと、パレットテーブルは、その出現順に、次のデータがセットされ、このパレットテーブルを用いて圧縮した画像データは、原画像の24ビット/画素に対して、2ビット/画素になるから、1/12に圧縮されたことになる。勿論、実用的な8ビットのパレットを用いる場合は、24ビット/画素に対して8ビット/画素になるので、情報量は1/3に圧縮されることになる。

【0089】

col[0]=(255, 255, 255): 圧縮コード“00”
col[1]=(0, 0, 0): 圧縮コード“01”
col[2]=(240, 64, 0): 圧縮コード“10”
col[3]=(128, 128, 255): 圧縮コード“11”
図22はこのようにしてパレット圧縮されたデータを伸長して出力する処理手順の一例を示すフローチャートで、圧縮データが入力されると伸長部2011が実行するものである。

【0090】図22において、まず、ステップS2601で変数iに零をセットし、ステップS2602で未処理のパレットデータがあるか否かを判定して、あればステップS2603へ進んでパレットデータcol[i]を取出し、ステップS2604で図20にS2203で示したのと同様の色合わせ処理を行う。続いて、ステップS2605で処理済みのパレットデータをパレットpcol[i]へセットし、ステップS2606で変数iをインクリメントした後、ステップS2602へ戻る。

【0091】未処理のパレットデータがなくなるとステップS2608へ進んで、圧縮データcodeに対応するパレットデータpcol[code]を出力し、ステップS2609で未処理の圧縮データがあるか否かを判定して、あればステップS2608へ戻り、なければ処理を終了する。

【0092】このように、本実施形態によれば、パレット圧縮されたデータを伸長する場合に、そのパレットテーブルのデータに色合わせ処理を施した後、圧縮データに対応する処理済みのパレットデータを出力すればよい。なお、未定義のパレットについては、色合わせ処理が不要であることはいうまでもない。例えば、8ビットのパレットテーブルによって圧縮した128×128画素の画像の場合、一画素ずつ色合わせ処理を行うと16,384回の演算が必要になるが、本実施形態においてはパレットの数に応じて最大でも256回で済ませることができ、演算時間を1/64以下に短縮することができる。

【0093】＜第5の実施形態＞以下、本発明にかかる第5実施形態の画像処理装置を説明する。なお、第5実施形態において、第3実施形態と略同様の構成については、同一符号を付して、その詳細説明を省略する。

【0094】以下では、既に出現した色を例えば三色記

18

憶し、注目画素の色が記憶した色と一致すれば予め決められた符号を出力し、どれとも一致しなかった場合は不一致を示す符号と注目画素の値(色)そのものを出力するとともに、記憶する色を更新する第三の圧縮方法により画像を圧縮したデータを伸長する例を説明する。

【0095】図23は第三の圧縮方法により画像を圧縮する手順例を示すフローチャートである。

【0096】まず、ステップS2401で記憶する三色の初期値をセットする。この例では、文書に多いと予想される白黒赤の三色を、それぞれ変数col[0], col[1], col[2]に記憶する。なお、この初期値設定は省略しても構わない。その場合は、処理が進んで最初の三色がセットされるまで変数col[i]はブランクのままになる。次に、ステップS2402で変数col[i]の初期順位を変数table[i]にセットする。この例ではtable[0]=0, table[1]=1, table[2]=2にする。次に、ステップS2403で未処理の画素があるか否かを判定して、あればステップS2404へ進み、なければ処理を終了する。

【0097】未処理の画素がある場合、ステップS2404で注目画素の値Dが設定した色col[table[0]]と一致するか否かを判定して、一致する場合はステップS2405に進み符号codeに‘0’をセットした後、ステップS2403へ戻る。

【0098】また、D≠col[table[0]]の場合はステップS2406へ進み、Dがcol[table[1]]と一致するか否かを判定して、一致する場合はステップS2407に進み符号codeに‘10’をセットし、ステップS2408でtable[0]=1かつtable[1]=0に変更した後、ステップS2403へ戻る。ここで変数table[i]を変更するのは、この圧縮方法が可変長符号を生成するものであり、col[table[0]], col[table[1]], col[table[2]]の順に1ビット、2ビット、3ビットの符号を割当てているためである。従って、col[table[0]]には、より出現し易い色をセットした方が効率的であり、そのために変数table[i]を変更する必要がある。例えば、ステップS2408の処理直前にセットされた色の順番が「白黒赤」だった場合、処理後は「黒白赤」の順番になる。

【0099】また、D≠col[table[1]]の場合はステップS2409へ進み、Dがcol[table[2]]と一致するか否かを判定して、一致する場合はステップS2410へ進み符号codeに‘110’をセットし、ステップS2411で上述したのと同じ理由からtable[0]=2, table[0]=0かつtable[1]=1に変更した後、ステップS2403へ戻る。

【0100】一方、変数table[i]のどれとも一致しなかった場合はステップS2412へ進み、不一致を示す符号‘111’と注目画素の値Dとを合わせた例えば27ビットを符号codeにセットし、ステップS2413で上述したのと同じ理由からtable[0]=2, table[1]=0か

19

つtable[2]=1に変更し、ステップS2414でcol[table[0]]に注目画素の値Dをセットした後、ステップS2403へ戻る。

【0101】以上の処理をすべての画素に施せば、圧縮した画像データを得ることができる。画像によってはステップS2412からS2414の処理が多くなり、圧縮結果のデータサイズが大きくなることもあるが、DTPによって作成された文書などは約1/20に圧縮されるのが一般的である。

【0102】図24はこのようにして圧縮されたデータを10を伸長して出力する処理手順の一例を示すフローチャートで、圧縮データが入力されると伸長部2011が実行するものである。

【0103】まず、ステップS2501で変数col[i]に圧縮時と同じ色の初期値をセットし、ステップS2502で図20に示したS2203と同様の色合わせ処理を初期設定した三色に施し、ステップS2503で色合わせ処理されたデータを変数pcol[i]へセットする。例えば、出力デバイスがカラープリンタで、col[i]に白黒赤の三色をRGBデータをセットした場合、pcol[i]にセット20されるCMYKデータは次のようになる。

```
【0104】 col[0] = {255, 255, 255}
col[1] = { 0, 0, 0}
col[2] = {255, 0, 0}
pcol[0] = { 0, 0, 0, 0}
pcol[1] = { 0, 0, 0, 255}
pcol[2] = { 0, 255, 255, 0}
```

ただし、圧縮時の説明で言及したが、この初期値設定は省略される場合がある。つまり、圧縮時に省略されていたら伸長時は設定しない、圧縮時に設定されていたら伸30長時も同値を設定することになる。

【0105】続いて、ステップS2504でpcol[i]の初期順位を変数table[i]にセットする。これも圧縮時と同様の順番にする必要があり、この例ではtable[0]=0, table[1]=1, table[2]=2にする。次に、ステップS2505で未処理の圧縮データがあるか否かを判定して、あればステップS2506へ進み、なければ処理を終了する。

【0106】未処理の圧縮データがある場合は、ステップS2506で圧縮データcodeが'0'かどうか判定す40る。'0'であればステップS2507へ進みpcol[table[0]]を出力した後、ステップS2505へ戻る。

【0107】また、code≠'0'の場合はステップS2508へ進み、圧縮データcodeが'10'かどうか判定する。'10'であればステップS2509へ進みpcol[table[1]]を出力し、ステップS2510で変数table[i]を変更した後、ステップS2505へ戻る。なお、変数table[i]の変更は、圧縮時と同様にしなければならないので、この例ではtable[0]=1, table[1]=0にする。

20

【0108】また、code≠'10'の場合はステップS2511へ進み、圧縮データcodeが'110'かどうか判定する。'110'であればステップS2512に進みpcol[table[2]]を出力し、ステップS2513でtable[0]=2, table[1]=0およびtable[2]=1に変更した後、ステップS2505へ戻る。

【0109】また、code≠'110'の場合はステップS2514へ進み、圧縮データcodeを3ビット左シフトして色データを取り出して変数xにセットし、ステップS2515でステップS2502と同様の色合わせ処理を変数xに施し、ステップS2516でその処理結果x'を出力する。なお、圧縮データから色データを取り出す際のシフト量は、前述した不一致を表す符号のビット数に応じて変化するものである。次に、ステップS2517でtable[0]=2, table[1]=0, table[2]=1に変更し、ステップS2518でpcol[table[0]]に変数x'の値、つまり色合わせ処理された新しい色データをセットした後、ステップS2505へ戻る。

【0110】このように、本実施形態によれば、第三の圧縮方法で圧縮されたデータを伸長する場合に、その初期値に設定される色データおよび不一致の符号に対応する色データに色合わせ処理を実行すればよい。従って、一画素ずつ色合わせ処理を行う場合に比べて、色合わせ処理の演算回数が大幅に低減できることは明らかであり、その演算時間を大幅に短縮することができる。

【0111】なお、上述では、記憶する色を三色にする例を説明したが、これに限らず、二色でも、四色でも、それ以上であっても構わない。また、記憶されたどの色とも一致しない場合に、どの色とも一致しないことを示す符号と、その注目画素の値（色データ）とを合わせて圧縮データとする例を説明したが、これに限らず、符号に添付する情報は色を表す情報であれば何でもよい。例えば、記憶された色との差分データを添付してもよいし、別途、パレットテーブルを作成してパレットコードを添付してもよい。

【0112】以上説明した様に、本発明の上述の実施形態によれば、圧縮画像を伸長して出力デバイスへ出力する際の処理を低減する画像処理装置及び方法を提供することができ、例えば、色変換処理等の処理を低減することができ、その処理時間を短縮させることができる。

【0113】なお、本発明は、ホストコンピュータとプリンタなどの複数の機器から構成されるシステムに適用しても、ファクシミリ装置のような一つの機器からなる装置に適用してもよい。また、本発明は、システムあるいは装置に媒体に格納されたプログラムを供給することによって達成される場合にも適用できることはいうまでもない。また、出力デバイスとして、レーザビームプリンタやインクジェットプリンタのほか、CRTやFLCDなどの表示装置も用いることもできる。

50 【0114】図25は、上述の第1～第5の実施形態の

21

アルゴリズムを実施する際のシステム構成の一例を示す図である。

【0115】図25において、1、2、3はホストコンピュータであり、後述の構成要素を含む。4、5、6はホストコンピュータ1〜3（I〜III）によって処理された画像データに基づく画像形成を行う出力部であり、4は多値カラーデータに基づいてプリントを行うカラーレーザビームプリンタ、5は2値カラーデータに基づくプリントを行うカラーバブルジェットプリンタ、6は強誘電性液晶を用いて表示を行うFLCDである。また、10

7、8、9はそれぞれ出力部4〜6（出力部I〜III）とネットワーク18をつなぐためのインターフェースである。

【0116】10はホストコンピュータのメインCPUであり、例えば、図1、図2、図11、図20〜図24等の各処理を実行する。11はCPUのワーキング領域として用いられるRAM、12はCPUのプログラム、OSを格納するROM、13はハードディスクであり、DTPアプリケーションソフトウェアや、これにより作成された文書画像データ、PDLコードから展開された20画像データ等を格納する。14はフロッピーディスクドライブであり、フロッピーディスク15に記憶されているデータをホストコンピュータ1に取り込むために用いられる。ここでフロッピーディスク15には、上述のCPU10が行う処理のためのプログラムを格納しておき、ここから読み出されたプログラムに基づいて上述の実施の形態の手順を実現してもよい。また、フロッピーディスクのかわりに、光ディスクや光磁気ディスクのような他の媒体を用いてもよい。

【0117】16は操作部であり（キーボードやマウス30からなる）、マニュアルでの指示、データの入力のために用いられる。17はモニターであり、文書画像データの作成等にも用いられる。

【0118】尚、他のホストコンピュータ2、3においても、ホストコンピュータ1と略同様な構成をしている。従ってその説明は省略する。

【0119】図26は図25のシステムにおいて、本発明の上述の実施形態を実現する際のCPU10の手順を示すフローチャートである。

【0120】先ず、ステップS1において、操作部1640からマニュアル指示により、出力先を指定する。この指定情報をCPU10が取り込み、通信先を指定する。次に、ステップS2において、設定された出力先に対してコマンドを送信し、ステップS3において、出力先からのステータスデータを受信する。このとき、ステータスデータの中には、出力先の装置の識別（プリンタ、モニター等）や、装置の特性（色再現特性（プロファイル））等を示す情報が含まれる。

【0121】ステップS4において、受信したステータス情報に基づいて、色変換パラメータ（例えば図4の色50

22

変換マトリックス等）を求めて設定する。ステップS5では、設定された色変換パラメータを用いて、上述の各実施形態のPDLデータの展開処理を実行する。ステップS6では、展開された画像データを出力先に送信し、ステップS7で、出力指示に応じて出力先での画像形成を行なわせる。

【0122】また、上述した実施形態の考え方では、CMS（カラーマネージメントシステム）にも適応可能である。

【0123】図27は、その例を示す図である。

【0124】先ず、3001において、DTPアプリケーションによる文書画像データを作成し、3002でOSに基づくPDLコード化を行う。これらに図13の場合と同様である。

【0125】次に、3003において、図25の出力部IIIのプロファイルを受信し、そのプロファイルに基づく色処理パラメータを用いて出力データに展開する。

【0126】そして、3004において出力データを出力部IIIに送信し、3005で出力部IIIによる表示を行う。

【0127】出力部IIIのディスプレイにおいて、PDLデータをプレビューした結果、希望の画像であると判断された場合には、次に出力部Iでハードコピーを作成する。

【0128】このとき、出力部Iと出力部IIIの色味をマッチングさせるため、出力部Iからのプロファイルを受信し、前述の出力部IIIのプロファイルとの双方に応じて色マッチングパラメータを作成する（3006）。

【0129】そして、3002でPDLコード化された文書画像を、色マッチングパラメータを用いて出力データに展開し（3007）、出力部Iへデータ送信し（3008）、出力部Iでプリントする（3009）。

【0130】以上のような手順により、色特性の互いに異なる出力部により同一の文書画像を出力させる場合に、出力画像の色味マッチングを行った画像データを高速に作成することができる。

【0131】ここで、色マッチングパラメータとして、例えば、図5のような色変換マトリックスを用いるときは、出力部I用のマトリックスを合作することによって色マッチングのためのマトリックスを作成することができる。

【0132】以上の例では、カラー処理を説明したが、本発明は多値白黒画像など白黒処理についても応用することができる。その場合には、上述の色処理パラメータとして色変換マトリックスのかわりに例えば、変換係数を用いてもよい。

【0133】また、色処理パラメータとしては、色変換係数に限らず、下色除去量や墨入れ量などの係数であっても良い。

【0134】本発明は上述の実施形態に限らず、クレー

ムの記載の範囲内で様々な変形、応用が可能である。特に上述した各実施形態を組み合わせても良いのは勿論である。

【発明の効果】以上説明した様に本発明によれば、色指定された文字コードや線画等の画像情報を可視画像出力装置に出力する際、その可視画像出力装置に依存したデータに効率良く、且つ高速に変換することが可能になる。また、その際に、複数の出力装置間の色合わせを行うことができる。

【0135】また、本発明によれば、所定画素ブロック10単位の符号化画像データがある場合において、可視画像出力装置に依存したデータを効率良く且つ高速に変換することが可能になる。

【0136】また、実質的に同じ色あいのカラー画像に関して、可視画像出力装置に適応するデータに変換する処理を実質的に行わず、効率良くその装置に適応したデータを生成することが可能になる。

【0137】更に、圧縮された画像を伸長して出力デバイスへ出力する際の処理を低減することができる。

【0138】また、更には、ネットワーク上に接続され20た複数のデバイスを用いた効率の良い画像処理方法が提供できる。

【0139】

【図面の簡単な説明】

【図1】第1の実施形態におけるデータの流れを示す図である。

【図2】第1の実施形態における処理手順を示すフローチャートである。

【図3】第1の実施形態における文字図形を含む画像の一例を示す図である。

【図4】第1の実施形態における文字コードに対するビットマップ展開及び色処理の原理を示す図である。

【図5】第1の実施形態における図形コードに対するビットマップ展開及び色処理の原理を示す図である。

【図6】第1の実施形態における生画像データの圧縮符号化処理の構成を示す図である。

【図7】異なる色による境界部分の画像状態を示す図である。

【図8】図7の画像データを周波数変換し、量子化した値を示す図である。

【図9】図7の画像におけるRGB直流成分とRGB差分直流成分の変遷を示す図である。

【図10】第1の実施形態におけるイメージ符号コードを伸長し、色処理を行う装置のブロック構成図である。

【図11】第2の実施形態における処理手順を示すフローチャートである。

【図12】第2の実施形態の効果を示すための画像データの一例を示す図である。

【図13】PDLコードを展開し、色処理する従来の処理工程を示す図である。

【図14】圧縮データを伸長してカラープリンタに出力する場合を説明する図である。

【図15】図14に示す色合部の処理を説明する図である。

【図16】黒生成処理を説明する図である。

【図17】本発明にかかる一実施形態の画像処理装置の構成例を示すブロック図である。

【図18】画像の一例を示す図である。

【図19】圧縮データの一例を示す図である。

【図20】第3の実施形態の伸長処理手順の一例を示すフローチャートである。

【図21】パレット圧縮による画像データの圧縮手順例を示すフローチャートである。

【図22】本発明にかかる第4実施形態の画像処理装置の伸長処理手順の一例を示すフローチャートである。

【図23】第3の圧縮方法により画像を圧縮する手順例を示すフローチャートである。

【図24】本発明にかかる第5実施形態の画像処理装置の伸長処理手順の一例を示すフローチャートである。

【図25】本発明を実施するシステム構成を示す図である。

【図26】本発明を実施する際の手順を示すフローチャートである。

【図27】本発明をCMSに応用した例を示す図である。

【符号の説明】

101 DTPアプリケーションによる文書画像データの作成

102 OSに基づくPDLコード化

103 文字図形／イメージ符号コードの判別

104 文字図形の色処理

105 イメージ符号コードの色処理

106 文字図形コードのビットマップ化

107 イメージ符号コードのビットマップ化

108 文字図形・イメージの合成

109 出力（プリント、モニタ表示）

401 文字コード

402 Character Generator

403 色処理変換

404 文字コードがビットマップ化された画像

405 色変換処理されたデータ

501 Vectorコード

502 Vector Generator

503 色処理変換

504 ベクトルコードがビットマップ化された画像

505 色変換処理されたデータ

601 周波数変換部

602 量子化部

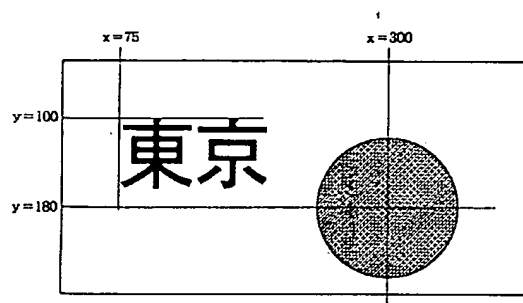
603 ブロック遅延部

604 符号化部

25

- 1001 復号化部
- 1002 交流成分検出部
- 1003 ブロック遅延部
- 1004 色処理部
- 1005 逆量子化部
- 1006 逆周波数変換部
- 1007 出力装置
- 1008 ブロック遅延部
- 1009 逆量子化部

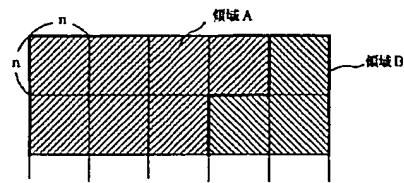
【図3】



26

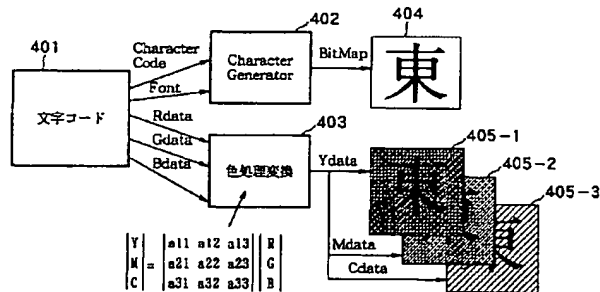
- 1010 逆周波数変換部
- 1011 色処理部
- 1301 DTPアプリケーションによる文書画像データの作成
- 1302 OSに基づくPDLコード化
- 1303 PDLコードのビットマップ化
- 1304 色合わせ処理
- 1305 出力 (プリント、モニタ表示)

【図7】

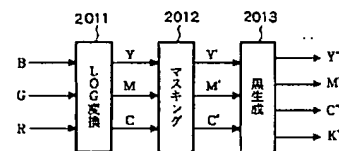
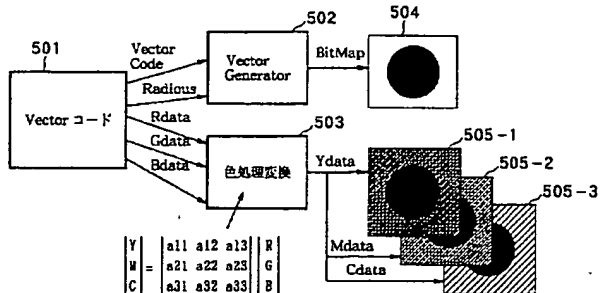


【図15】

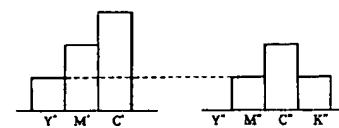
【図4】



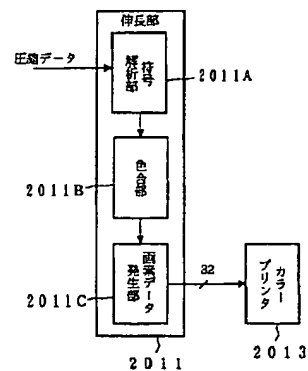
【図5】



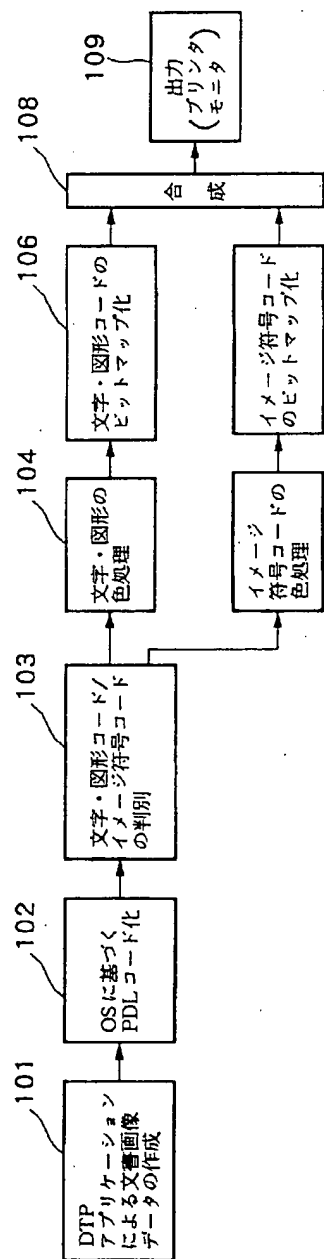
【図16】



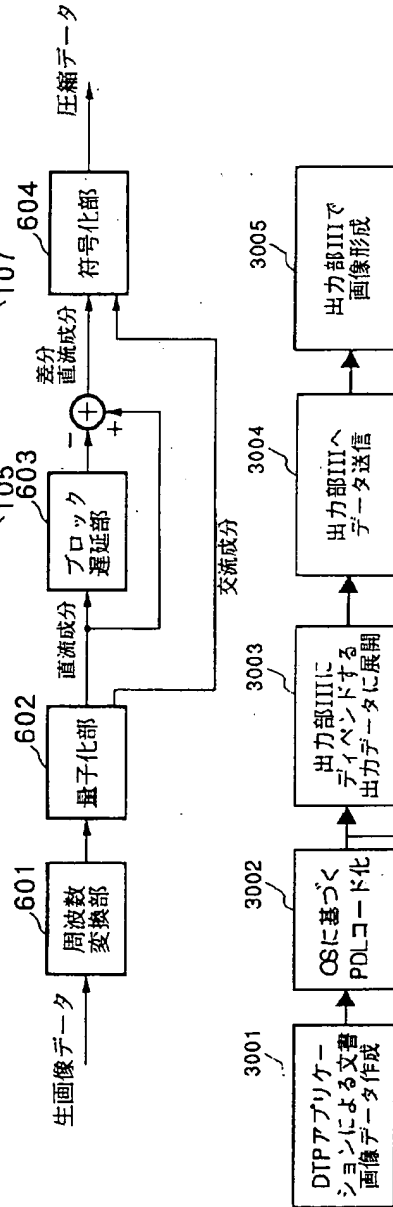
【図17】



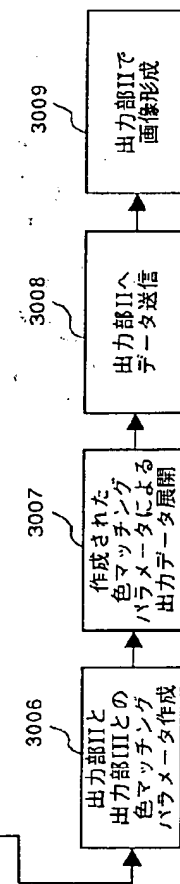
【図1】



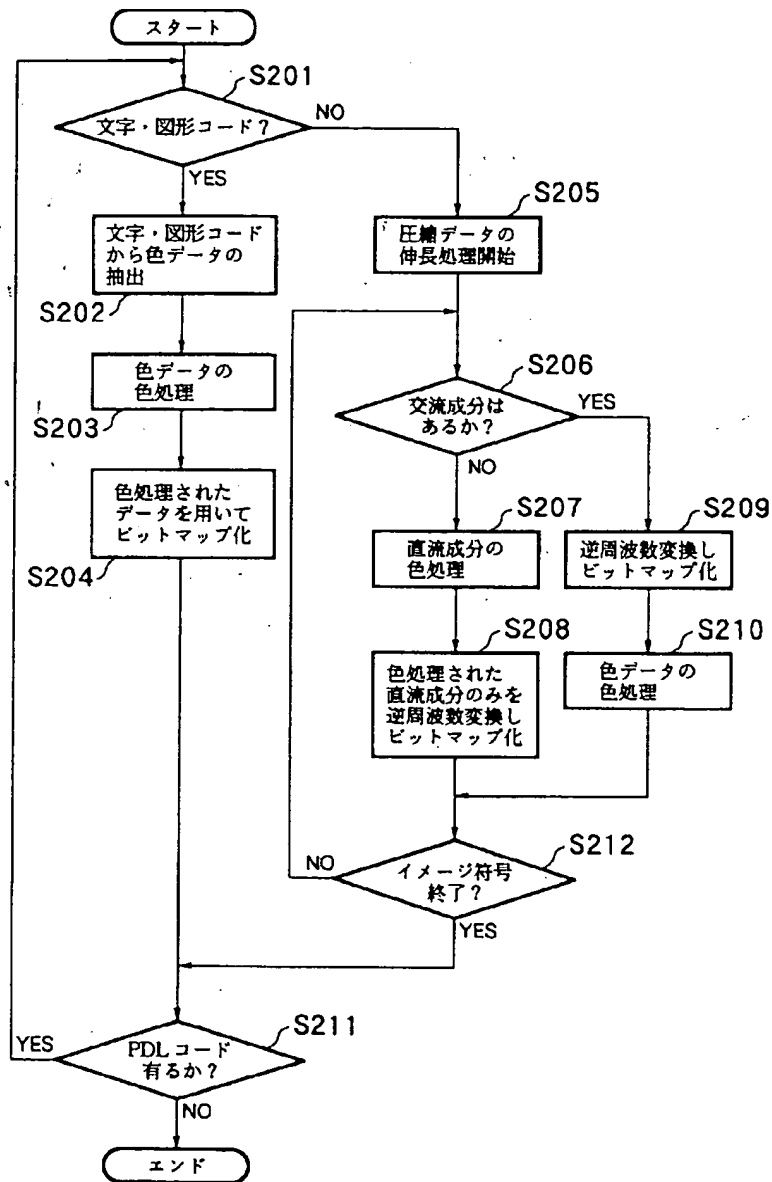
【図6】



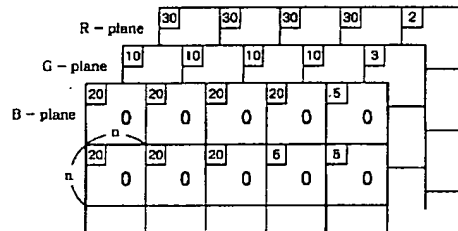
【図27】



【図2】



【図8】



【例 12】

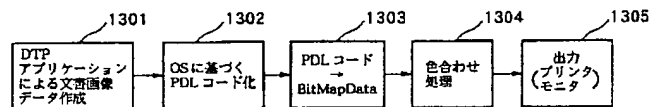
[illegible]

【图9】

RGB 直流成分 $\begin{pmatrix} 30 \\ 10 \\ 20 \end{pmatrix} \rightarrow \begin{pmatrix} 30 \\ 10 \\ 20 \end{pmatrix} \rightarrow \begin{pmatrix} 30 \\ 10 \\ 20 \end{pmatrix} \rightarrow \begin{pmatrix} 30 \\ 10 \\ 20 \end{pmatrix} \rightarrow \begin{pmatrix} 2 \\ 3 \\ 5 \end{pmatrix} \rightarrow \begin{pmatrix} 30 \\ 10 \\ 20 \end{pmatrix} \rightarrow \begin{pmatrix} 30 \\ 10 \\ 20 \end{pmatrix} \rightarrow \begin{pmatrix} 30 \\ 10 \\ 20 \end{pmatrix} \rightarrow \begin{pmatrix} 2 \\ 3 \\ 5 \end{pmatrix} \rightarrow \begin{pmatrix} 2 \\ 3 \\ 5 \end{pmatrix}$

RGB 差分直流成分 $\begin{pmatrix} 30 \\ 10 \\ 20 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \rightarrow \begin{pmatrix} -28 \\ -7 \\ -15 \end{pmatrix} \rightarrow \begin{pmatrix} 28 \\ 7 \\ 15 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \rightarrow \begin{pmatrix} -28 \\ -7 \\ -15 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$

【例 13】



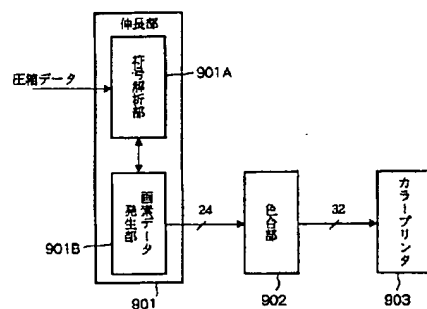
【图 19】

```

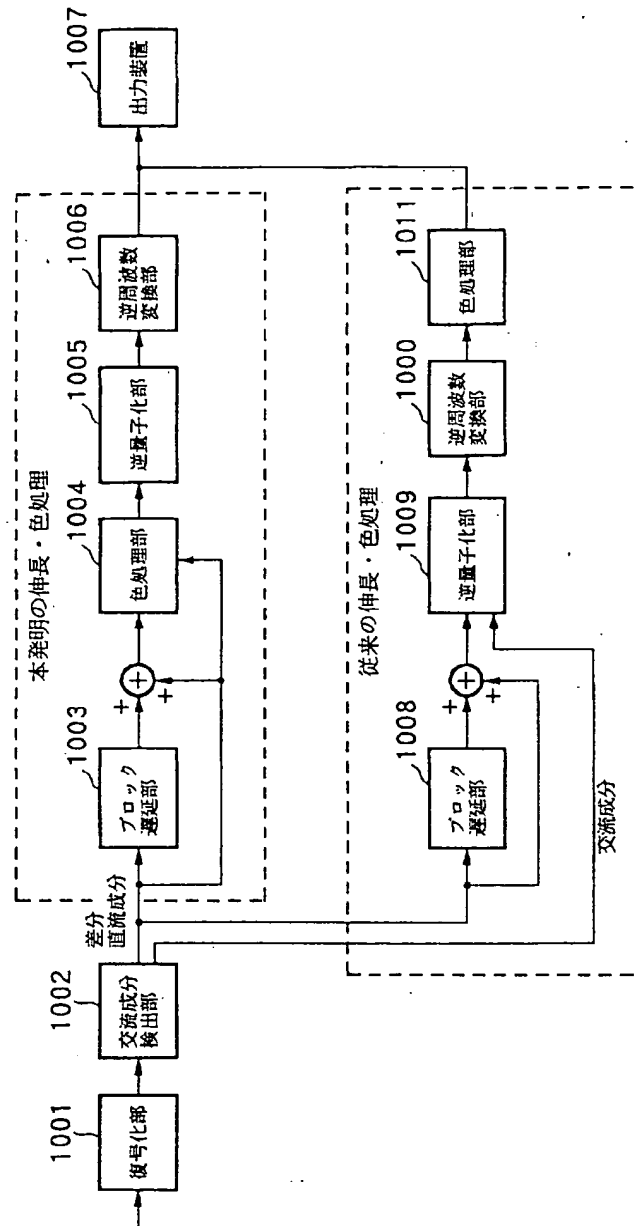
11111111111111111111111100011000
11110000010000000000000000000000
11111111111111111111111100100000
1000000010000000111111100001100
11110000010000000000000000000100
100000001000000011111100001000
000000000000000000000000000000100
100000001000000011111100010100
11111111111111111111111100001100

```

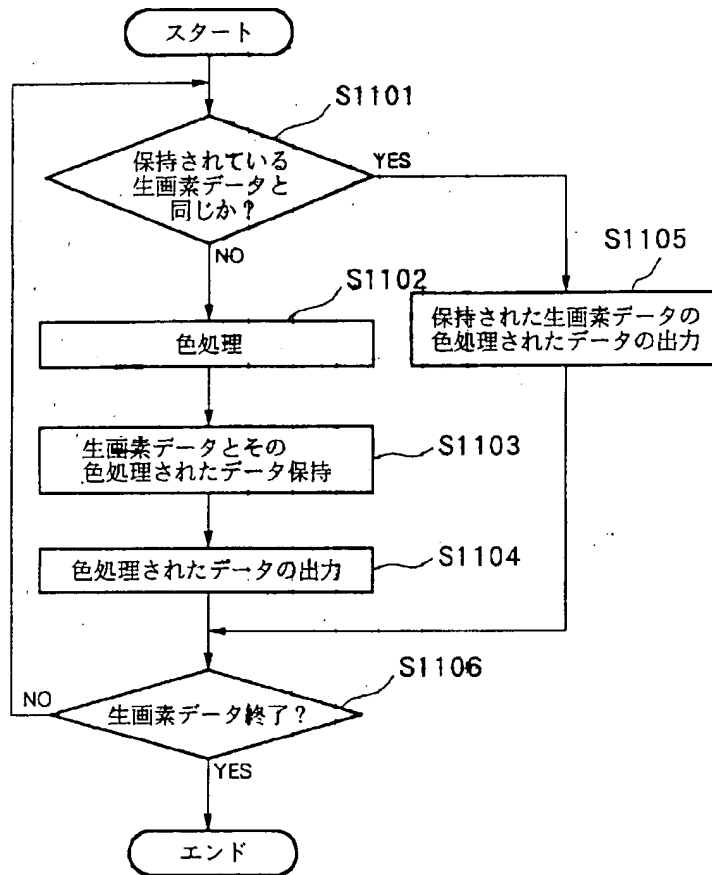
【图 14】



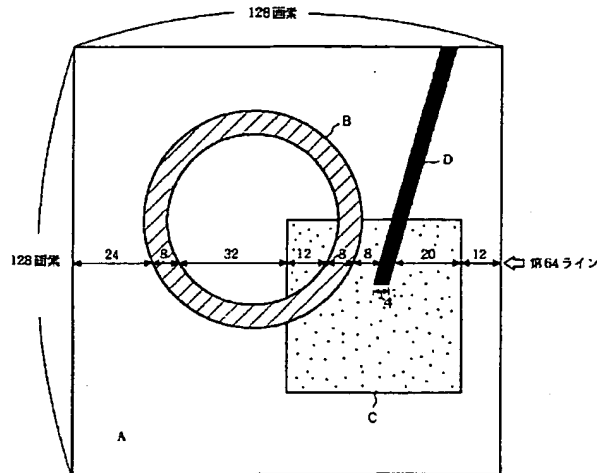
【図10】



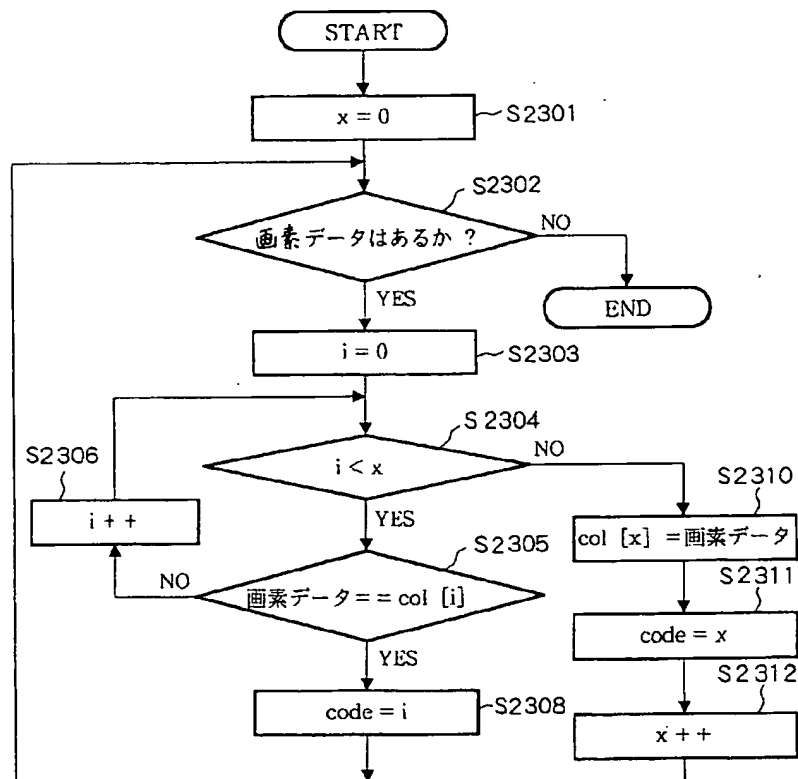
【図11】



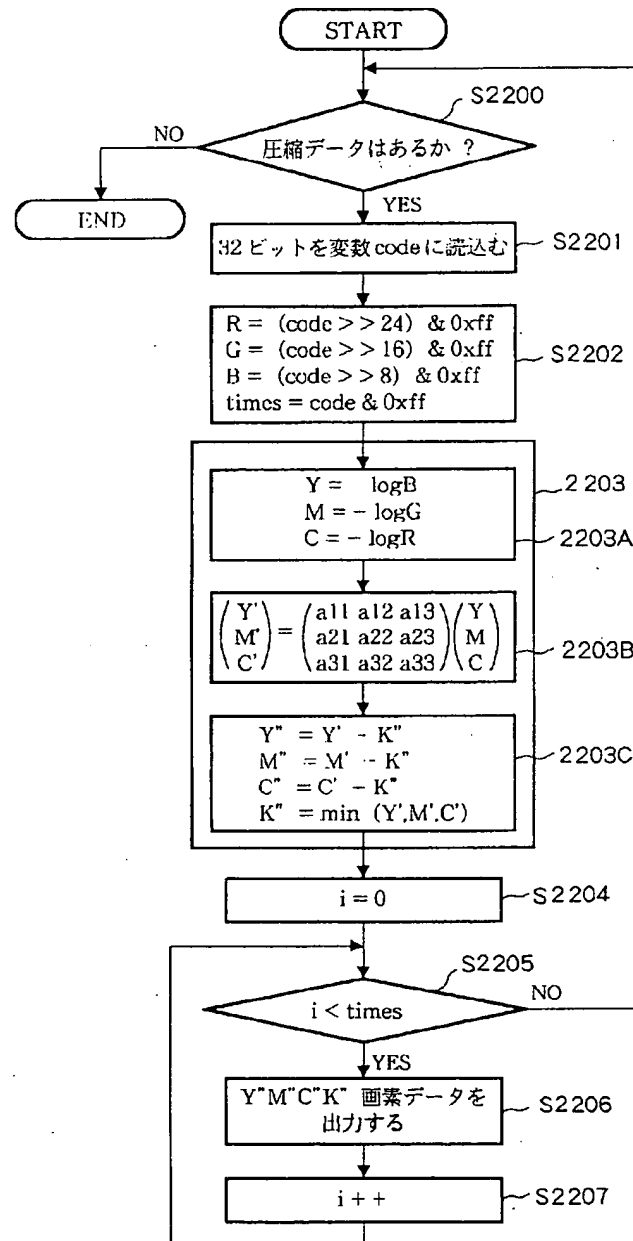
【図18】



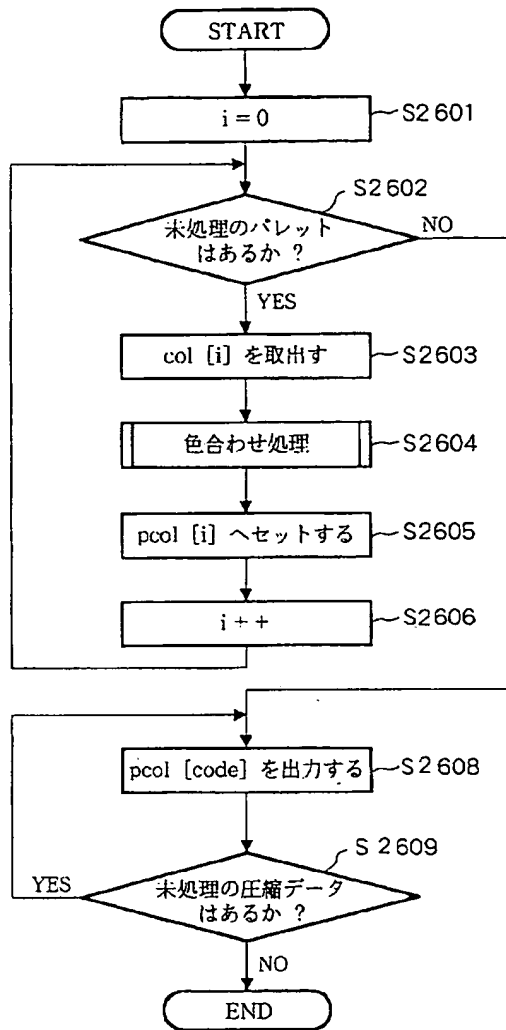
【図21】



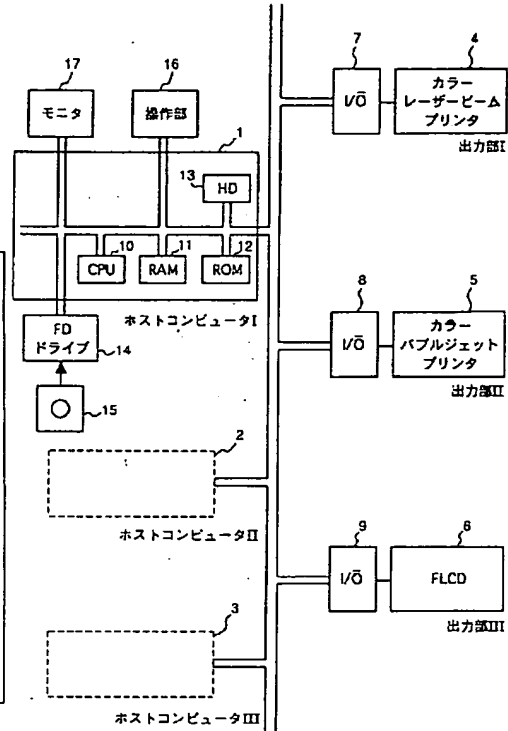
【図20】



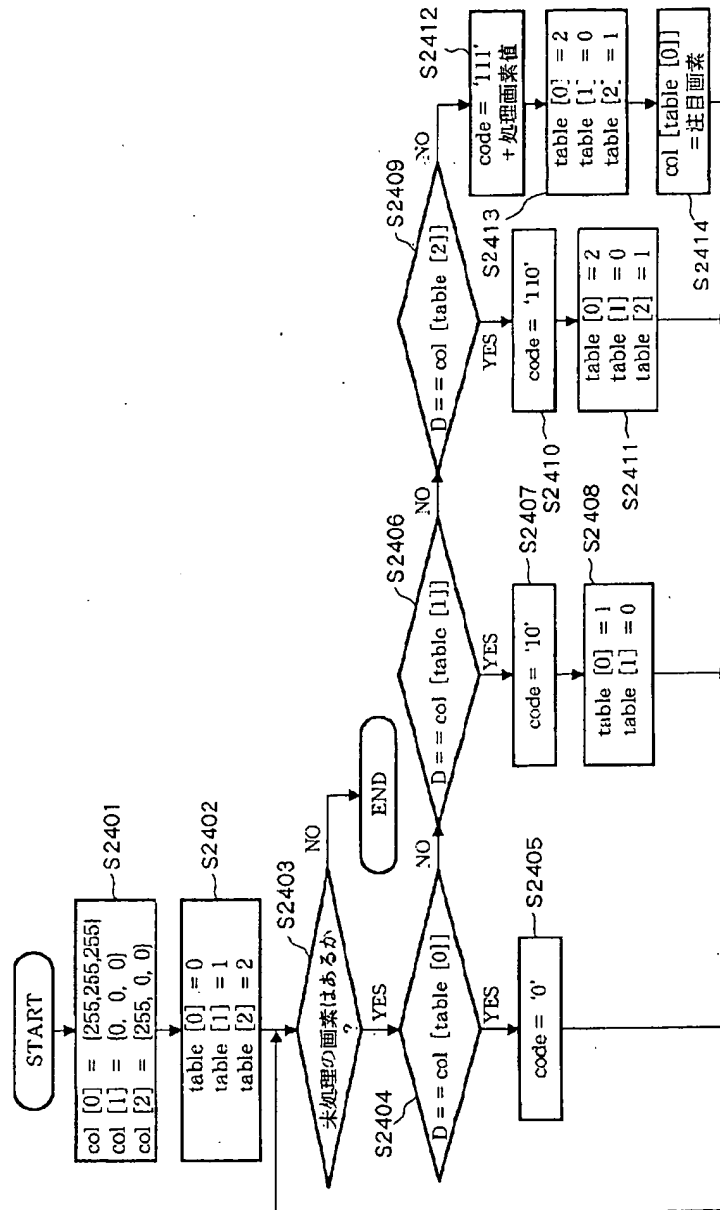
【図22】



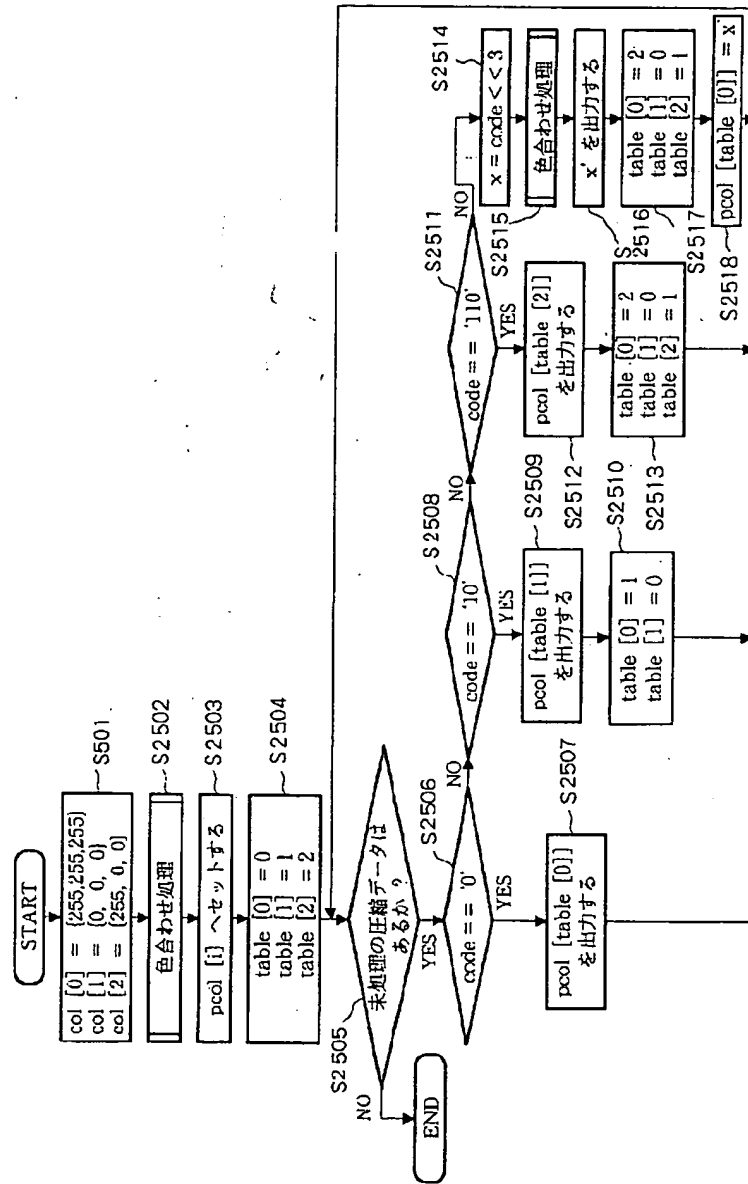
【図25】



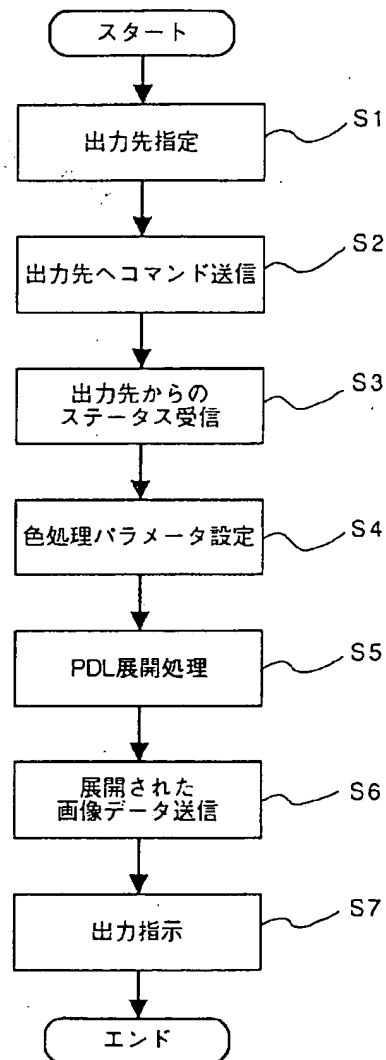
【図23】



【図24】



【図26】



フロントページの続き

(51) Int. Cl. 6

識別記号

庁内整理番号

F I

技術表示箇所

// G 0 6 T 9/00

THIS PAGE BLANK (USPTO)